THE APPLICATION OF COMPUTER
CONTROLLED DATA MANAGEMENT
TO THE
HISTORIC STRUCTURE REPORT

Raymond L. Tschoepe

A THESIS

in

Historic Preservation

Presented to the faculties of the University of Pennsylvania in
Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

1993

John Milner, Lecturer, Historic Preservation

David G. DeLong, Professor of Architecture, Reader

David G. DeLong, Professor of Architecture
Graduate Group Chairman

PURPOSE STATEMENT

The purpose of this study is to introduce the design of a graphic-based computerized data management system designed to facilitate information storage and retrieval for the historic preservation/conservation practices with particular emphasis on its application to the Historic Structure Report.

INTRODUCTION

In his book "The Day the Universe Changed" (and PBS television series) author James Burke writes,

> One major result of printing was the emergence of a more efficient system of filing. With more than a thousand editions reproduced from the same original, book-collecting became fashionable. These collections needed to be catalogued. Moreover, printers had begun to identify their books by title, as well as author, so it was easier to know what a book was about.
>
> ...The new interest in indexing led to more factual analysis of the older texts. ...The new availability of data and the novel concept of information as a science in itself made the collation and use of data easier than before.[1]

The quest to impose order on bits of information whether they be stories, philosophical ideas or pictures has been the quest of each generation since the development of moveable type. Indeed, library science has achieved a high level of sophistication over the past 40 years due largely to the development of more advanced theories and methods of organization. The greatest changes have occurred over the past 10 years with the introduction of computerized cataloguing systems.

---

[1] James A Burke, *The Day the Universe Changed*, Little, Brown and Company, Boston, 1985, p. 120-121.

A number of attempts at cataloguing architectural features have met with varying degrees of acceptance. Some of these techniques are currently being assessed by HABS and HAER.[1] These require the imposition of symbols between the data and the researcher and they must be learned in order to make the system work efficiently. As the amount of data increases, the number of symbols needed to describe a feature and a location becomes increasingly cumbersome until a point is reached at which the symbol storage, manipulation and cross referencing begin to demand more memory than the stored information.

In this paper, I propose the storage and retrieval of information using a fundamentally graphic based electronic environment. In this way, the symbols needed to direct research are few and for the most part intuitive by nature (and tradition) such as the use of arrows to indicate direction.

In the preparation of a historic structure report (HSR) the preservationist/architect (P/A) is called upon to assemble hundreds of bits of grouped information. Generally speaking the information within groups is often related by intuitive hierarchies, but the relational links between groups are often linked only by cumbersome and vague indexing methods at best. Clear relational links throughout the document are necessary if a total "picture" is to be developed by the owner/client of the structure under study. For this reason, I have developed a computer model of a relational database that organizes all of the material generally assembled for an HSR into informational webs that allow for a complete cross-referencing within the report.

---

[1] Robert J. Kapsch, "HABS/HAER: A User's Guide," APT Bulletin, vol. 22, no. 1&2, 1990, pp. 22-34.

DEVELOPMENT OF THE MODEL

A conventional written report, such as an HSR, has the advantage of being accessible to anyone who has the physical skills necessary to turn pages and the mental agility to read. This written format is attractive chiefly due to its ease of use. Rarely is this simple format given second thought unless there is a proposal to change it. In this report I do propose a change to this elemental means of communication simply because the format which makes a written work so convenient and endearing is also the format which constrains its development.

Already the world of Compact Disc - Read Only Memory (CD-ROM) has entered into the instructional market. After totally revolutionizing the audio market, undergraduate textbook publishers have begun to implement a book-on-disc format to take advantage of the emerging computer technology. This offers the publisher the ability to revise and update with previously unimagined flexibility. The heart of the attraction to this technology is not the technical and marketing concerns of the publisher but rather the enhanced learning potential for the student. CD-ROM formats allow the author "to pre-connect" groups of related material so that relationships and analogous systems are more apparent to the student accustomed to reading from "start to finish" with relational divergence only referenced in footnotes and bibliography. This new format even allows the addition of video footage to a simple 4 inch disc. But, the price for these advantages is very high for some. Generally it means the *partial* abandonment of a form of data presentation that has been used for thousands of years. This long-lived method of opening a book and reading can be replaced by an extraordinarily complex substitute

for binding, paper and ink. Lest I overstate, I do not think anyone believes that books will be replaced in the foreseeable future, yet I think that there is little doubt that instruction and other data presentation channels will be supplemented to a great degree by emergent computer systems.

Where the CD-ROM is an unalterable final product (at this writing Read/Write CDs are in limited use) another level of media exists that is more familiar to anyone who has had even modest experience with computer technology. Most software programs, whether they are data management or data processing, give the operator the ability to store, retrieve and delete vast amounts of data. Unlike the CD-ROM which can only respond to data retrieval commands, simple programs permit the manipulation of information within a set of limits. Simply stated, the programming language, the ambient hardware, the programmed codes and the amount of memory restrict the user of traditional programs. However, with conventional (average) computer systems these limits are not ordinarily encountered.

The computer preparation of data that is often assembled for historical architectural surveys (HSRs) requires a number of conventional programs to be used. At its most basic level a program that will process text information (histories chain of title etc.) and one that will handle graphics (maps, elevations, plans etc.) are a minimal necessity. Many preparers of HSRs have incorporated word processing software for more than a decade now, and the use of sophisticated computer aided design (CAD) techniques is well known. The interaction between the two is extremely limited, often non-existent. This is one of the limits that is often reached with conventional programming. The final product of these efforts is usually...a book. A well formatted and attractively designed book, but a book at any rate.

The next level in the development of this model then is the combining of both text and graphic media in the same programming format. At present time this is most easily handled by a programming language called Hypermedia.

John Scully, current CEO of Apple computer defines it thusly:

> "...In broad terms, hypermedia is the delivery of information in forms that go beyond traditional list and database report methods. More specifically it means that you don't have to follow a predetermined organization scheme when searching for information. Instead you branch instantly to related facts. The information is eternally cross-referenced, with fact linked to fact, linked to fact.
>
> Hypermedia is particularly true to its name when it links facts across conventional subject boundaries...."
>
> "Hypermedia is not an application...rather it is a software engine."[1]

HYPERCARD

The hypermedia described by John Scully is expressed through the programming language of HyperCard. Designed and written by Bill Atkinson of Apple Computer, he describes it as "an authoring tool and an information organizer".

HyperCard can best be thought of as an information handler. A basic understanding of the programming platform is essential to maximizing the benefits you can realize from operating the system. Generally speaking, the entire system is organized around certain visual components which may or may not be activated and programming codes which run in the background in order to affect certain pre-determined actions. This, of course, is very much like any other computer program. For example, when you are typing a letter in a word processing program, the visual component may be as simple as the

---

[1] Danny Goodman, *The Complete HyperCard Handbook*, 2nd ed., Bantam Books, Toronto, 1988, p. xvii.

image of a sheet of paper. Typing the letter "o" closes a single switch which tells the computer how to shape the letter on the screen and where to position it. The process by which this is done remains in the background. HyperCard takes the essentials of programming and places them at the fingertips of the enduser, not the software developer. Programming in HyperCard is called Scripting. This paper will not go into the details of scripting since it is the topic of a variety of manuals and books. Suffice it to say that well known (to programmers) languages such as *COBOL* and *BASIC* are employed in the actions of HyperCard, but they are employed through a interpreter called HyperTalk. Think of it like a traveler preparing to journey to Greece. He can prepare for many hours in advance by learning and practicing Greek, or he can simply hire an interpreter at the last minute. HyperTalk is the interpreter. It is designed to read as close to conversational English as possible without losing necessary precision. Unlike standard programming languages, Hypertalk commands do not have to adhere to the extraordinary rigidity of computer language. It can, for example, ignore articles (a, the) forgives commas and allows the interchangeable use of certain words (in, into). This is not to say that it does not have its own idiosyncrasies and unbreakable rules, its just that it has far fewer. This programming shell then opens the door to the relational information handling that has been discussed so far. With Hypertalk, fairly sophisticated *custom* programming is available to the average user. Information can now be assembled and related in a goal specific direction.

Beyond scripting and Hypertalk, most users of Hypercard designed programs will become familiar with the visual organization of the system. Hypercard treats each screen image as a discreet piece of information. The size

and shape of each screen image gives the appearance of an index card (hence the name). Consequently each block of information thusly organized is called a "card".The information can be textual or graphic (or even photographic) or both. The screen can accommodate very large amounts of information.
In many cases the information can be greater that the visual area that can be viewed. Think of this as being able to write on the back of the index cards, and in some cases even on the edges. In Hypercard, related (or in some cases unrelated) cards are organized into groups called "stacks". A stack of cards can contain a single card or many hundreds. In organizing an HSR for example, one stack may contain all of the historical information, one may contain all of the images, one may contain future restoration plans, one may contain all of the analyses etc. Or, everything can be contained in one stack. By splitting the stacks though, related material is more easily organized, revised and amended. Stacks of information distributed (or sold) and intended to operate under HyperCard are known as "stackware" (as opposed, of course, to software).

As described above, each card can contain certain types of information. The textual information is contained in a "field". Field size, shape and the font within it is defined through Hypertalk. Therefore the field can be the size of a postage stamp or it can completely fill the screen. Adding text into a field is very much like working within a conventional word processing program that allows the basic carriage returns and character formatting.
When artwork is added to a card (usually a blank screen unless designed otherwise) it is displayed as a bitmapped image which can be altered under the usual options afforded by "paint" programs. This includes image enhancement, the addition of screening, deletion of all or part of an image

and relocation of all or part of the image. Certain other "special" effects such as rotation inversion etc., are also available. Text can be added to the card in direct conjunction to the image without the benefit of a field. The text added however does not respond like text in a word processing environment, but rather behaves like a piece of "art". It can be cut, rotated, distorted etc. Generally speaking, text added without a field is usually added as a label or directional guide.

Most users of Hypercard will generally be most familiar with added "buttons". A button is a resizeable moveable area (not unlike a field) that is defined through HyperTalk. A button is a "sensitive area" that when a cursor or other mouse guided icon is "clicked" within the area it can initiate an action. This action is defined by scripting attributed to the button area. For example it can initiate the movement from one card to another, or any one of hundreds of actions that can be defined through HyperTalk including playing music, showing a video or animating a process.

With a working knowledge of buttons, fields, cards and stacks information can be dismantled and reassembled into discrete units. These units can then be inter-related in a very specific fashion. Scripting (programming action) can be applied to cards and fields as well as the more commonly used buttons so that the range of interconnections is extremely variable.

When an architect/preservationist delivers a completed HSR to a client, there is no need to explain the mechanics involved in perusing the document. As stated above however, the usefulness of this electronic document which may contain not only basic information but also entire videos of the restoration process as well as dictated information, requires that we move beyond the constraints of bound paper.

This next step requires the computer manipulation of information through computer. Hypermedia is currently the simplest and most economical form of programming capable of handling this data. Developed for the Macintosh computer, HyperCard is the programming platform of choice. Due to its complexity and its graphic orientation, there are no comparable platforms designed for other operating systems - to date. Beyond HyperCard's capabilities, there are other reasons to recommend this system. First, HyperCard is very inexpensive. In fact, it is supplied with all Macintoshes purchased after August 1987. Because of this it is also in widespread use. This maximizes the HyperCard-literate pool toward which stackware can be directed. A pool of people that do not require extensive training seminars. This quality alone recommends it highly against customized "high-end" software programs that can take years to develop and debug. These high-end programs are often very expensive and of limited use to anyone who does not possess the means to purchase equipment that can run the software. In addition, custom tailoring of the software is almost impossible without extensive re-programming. Of more practical value, if a custom program is written for the development of reports such as HSRs

license is not granted to all endusers except through verifiable purchase. This can be extremely costly. Apple Computer Corporation does license HyperCard but does **not** claim license to the data that can be displayed on it. Therefore, stackware can be duplicated and distributed freely.

The availability and low cost of HyperCard and the duplication rights of associated stacks assure that the developed program will be used and understood by owners of the smallest museum houses to the largest estates and public buildings. In addition (and often most importantly), the hardware requirements are moderately priced. For this reason, I have chosen to design the electronic historic structure report (EHSR) on an Apple Macintosh model SE (model 68000 microprocessor running at a modest 8 MHz) with a hard drive and 4 MB of RAM memory. By all accounts this is a low to middle capability machine. Designing at this level insures that the data will be available to all.

HyperCard and the HSR

The model for this study was developed around the available information collected during the study of the Muhlenberg House located at 201 Main Street in Trappe, Pennsylvania. This structure was erected around 1755 by Jacob Schrack and was purchased by Reverend Henry Melchior Muhlenberg on Jan. 1, 1776. He lived there until his death in 1787. Among other things, Reverend Muhlenberg is known as the Patriarch of the Lutheran Church in America.

In the almost 250 years that the property was erected, it has undergone a number of major changes which include raising the roof to add an

additional half-story, as well as the removal of most of the fireplace masses and associated chimneys. In addition to this, a pent roof was removed and a full porch added. Many other changes have been documented and these are referenced in the HSR currently being assembled. This property provides an excellent resource to demonstrate the capabilities of electronic data manipulation.

Data collection on this property began in 1990 and continues to the present (April 1993). In the midst of ongoing restoration work, clues to its past form continue to come to light.

Under normal circumstances, this constant documentation of newly uncovered clues could become a source of frustration for the preservationist. It is usually advantageous for the property owner to have a "complete" HSR in order to solicit funding and provide a plan for restoration. During the early demolition phases of restoration, many bits of evidence supporting or questioning the currently accepted plan may be found. Documentation of the evidence is essential and alterations to the restoration course may be warranted. The preservationist must then consider means to incorporate this new data into the HSR while retaining financial responsibility for the re-production of what have become rather weighty tomes indeed.

At the annual Association for Preservation Technology conference held in Philadelphia (Sept. 23 - 26, 1992), many hours were given over to the discussion of just such matters of HSR preparation. Mary B. Dierickx of Architectural Preservation Consultants in New York, pressed for a standardized outline and stressed that the HSR should be prepared as early as possible *before* construction. This, she admitted, would leave an incomplete study, but felt as did others that a series of addenda could be added as

completion reports, recommendations for further study, maintenance plans, etc. To help to assuage the growing cost of HSR's, Andrea Gilmore of The Society for the Preservation of New England Antiquities Conservation Center in Waltham, Massachusetts, explained their use of multi-level reports.
A Level I study is prepared for "homeowners" scale buildings and would run between 5-10 pages. A Level II structure is defined as one of moderate size and limited complexity. These studies would typically be in the range of 40-45 pages. A Level III study would be a full scale HSR. Martin Jay Rosenblum of Martin Jay Rosenblum Assoc. located in Philadelphia, Pennsylvania, felt the cost could be reduced by truncating the process. A survey of his past clients revealed that they considered the interpretation of the structure to be the most important, the restoration plan itself, less so, and the maintenance plan the least important. As a matter of fact, the maintenance plan was ranked a distant third. Finally, John G. Waite of Mesick, Cohen and Waite located in Albany, New York suggested in the face of calls for standardization that the HSR be *flexible.*

HSR's truly demand flexibility and a certain degree of open-endedness. Yet, in honoring client demands there is an overwhelming urge to complete the report, bind it and present it as the "last word" on the property. For a few simple structures, this may be the case. However, historical surveys, like science are disciplines of *constant* discovery. As investigative techniques are refined new answers (and new questions) come to light. Sometimes these "new" answers challenge our firmly held past notions. Consequently, there should be a vehicle that allows us to present data with an understanding that our knowledge is incomplete, but this vehicle should provide us with a means for completing it.

The vehicle that best handles this flexible open-ended reporting is electronic data manipulation. Specifically, for the purposes of this study, the open architecture of HyperCard is especially well suited.

DESIGN

To begin the preparation of data for the computer, "packages" of information for entry are assembled. A well organized table of contents (TOC) is the heart of arranging these informational groups. The TOC of the Muhlenberg House HSR is organized into three volumes. The first volume consists of 4 "chapters". These are an Introduction, Methodology , Description of the Site Prior to Investigations and a Historical Background. Each of these chapters is composed of a number of sub-heads which introduce related topics. Volume II contains one chapter which deals exclusively with Architectural Investigations. This chapter stands in contrast to the preceding four in that most of this material is illustrative in nature, while the majority of the previous four is textual (with photos). The final volume (III) consists of four chapters detailing Materials Analysis, Archeology, Conclusions, Recommendations, Notes and a Bibliography. Also in this volume is a collection of Appended material such as the Chain of Title, Chronological list of Owners and Occupants, Architectural Drawings, List of Artifacts, Lab Reports etc. This volume is a mix of topics as well as information formats, so that there is a blend of graphic and textural information. Computer data management excels in the management of volumes like this. The programming is able to provide the links between appendicular material and the main body of the report in a seamless unobtrusive fashion.

Computer data management systems such as HyperCard allow for the almost unlimited addition and revision of data. Nothing is ever addenda, but simply an information group that can be tied into the body of the report wherever it is relevant.

When the information groups are assembled, each related piece of information can be input to the computer into "stacks" of material. A stack in HyperCard is analogous to the card file drawer in libraries. One drawer may contain all of the material in volume I another, all of the material in volume II and so on. Finally, and the approach that is used in this project, all of the information may be contained in a single drawer with heading cards placed along its length to segregate groups of information.

Card Design

Each card in the drawer (think of index cards) is initially a blank card or, in this case, a screen. Cards can be sized within limits so that more than one can appear on the screen at any one time. For most uses the card size should match the available space. In this case a standard Apple Macintosh 9" screen is used.

Each card has at least two layers that can be modified with standard "paint tools" to configure spaces that will house information most efficiently. When designed, changes made to the background layer of the card will be reflected on each card throughout the stack. By analogy, a card file drawer may contain 250 cards. If I were to stamp "Muhlenberg House" on the upper right hand corner of each card in the drawer, this would be an identifying imprint of the data collection. No matter what was subsequently typed on the card, the

imprint would identify it as one of a group of related cards. I might also draw in a square in the upper left hand portion of each card which, although blank now, will be labeled differently as data is added. This label will act as a reference key, or better, a cross-reference key. Each card then throughout the stack will have a similar appearance. (Fig. 1)

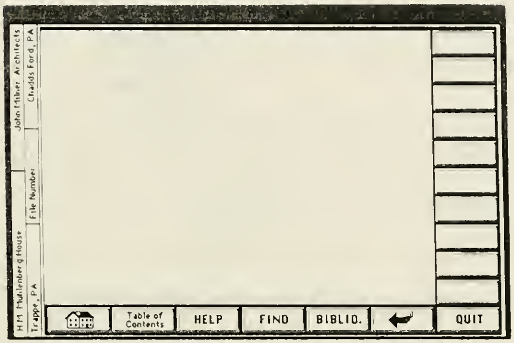
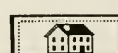
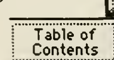


Figure 1. The basic card displaying "background" buttons.

The left side of the card lists the project, the architect and other relevant information. The central rectangular portion of the card is blank. This is where data, both textual and/or graphic will be placed. Finally, a series of small rectangles surrounds this central area on two sides. These areas have been reserved for active areas, "buttons". The horizontal row of buttons (on the bottom) have been programmed to perform a set of functions that are useful when navigating throughout the entire stack. For this reason, the programming for these has been applied to the card background so that they appear on all cards. As you can see from the above, the buttons perform certain basic functions. For instance, clicking-on this  will take the viewer back to the first screen. This button 

as it suggests, takes one to the table of contents. This is a central switching point. Once at the table of contents, new topics can be explored.

The HELP HELP button does just that. Clicking here gives the user some basic information about the program and its navigational features. It is useful when maneuvering through the screens has become confusing for one reason or another, or when simply more operational information is sought. The Find FIND button acts like an index. Clicking "FIND" activates a mechanism for searching for a particular word or group of words. The next button BIBLIO. keeps the bibliography handy. A left facing arrow in the next button  is particularly useful in that it performs the function of "go back". In other words it returns the user to the last screen viewed. Finally, the last button QUIT is self explanatory. The program can be exited at any time. This is analogous to simply closing the book.

The vertical column of buttons on the right are different in that the functions that they perform are card dependent. Consequently, they are written on the foreground of the card and are quite variable.

When information is added to the central area of the card it can be added through a number of input routes. If the information is textual, a "field" is placed in the area. A field in HyperTalk is a re-sizable rectangular area in which type fonts and leading can be specified. When this is determined, the text information is simply typed into the field as you would into a conventional word processing program. Type can be altered to reflect many styles, sizes and variations beyond those originally specified. For instance, headings and sub-headings may be placed in a larger sans-serif font while emphasis can be achieved through the use of bold or italic settings.

When the field on a particular card has been filled, a new card is created with a new field and typing continues. At this point it is necessary to devise a technique that allows the reader to "turn pages". On each "page" of

text, a pair of buttons is included that when activated, i.e. "clicked on", with the mouse icon it permits the reader to page forward or backward. (The mouse is a manually controlled input devise that positions a cursor on screen) In this project left and right facing triangles (arrows) serve this purpose.

Historic Structure Reports, when prepared properly, include footnote marks throughout the text that verify factual information by citing source material. Notes are usually placed at the bottom of the page (footnotes) or just as likely at the end of the report. Either of these arrangements are awkward in the HyperCard format, however, the programming system allows for the placement of footnote information in windows of "hidden" text directly on each page (card). Reference information is indicated by a tiny square that when clicked-on opens a window with the reference information. Another click on the square hides the window. (Fig. 2 - 3)

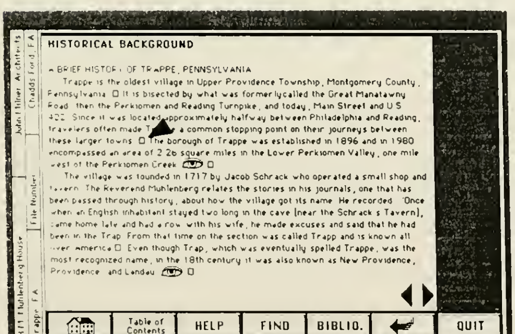


Figure 2. The arrow indicates the tiny square "button' that "hides" the relevant note.

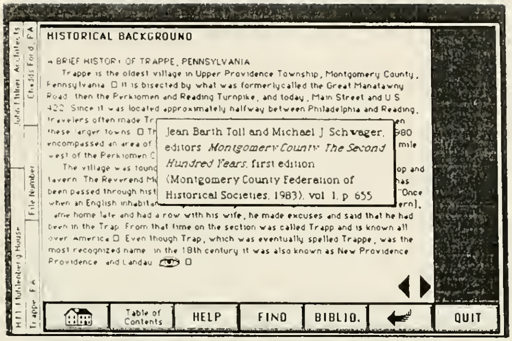


Figure 3. When activated, "clicked", the note material is displayed. A second click hides it again.

In addition to the squares, text material will usually make reference to photographs or other illustrative information. In this program figures are indicated by a small eye.

Clicking-on the eye will place the figure on the screen. A second click on the "return arrow"

brings the reader back to the text. This system makes it unnecessary to label figures 1, 2, 3, etc. or even footnotes for that matter. Each is linked directly with a line of type so that there is no need to search for each block of information.

Illustrative and photographic information is usually entered directly into the blank central field of each card. Once entered, it can be manipulated by a number of "paint" tools. Screens can be added, figures can be reversed, cut and moved, to name a few of the manipulations available. The route of entry for graphic material is usually through an image scanner. A scanner converts an image into a series of dots or pixels that can be translated through the computer and placed on the screen. The image is usually not placed directly into HyperCard, but rather into a program that allows for a great deal

of adjustment to the scanned image. Currently, programs such as Adobe *PhotoShop* and Letraset *Image Studio* are used to receive the raw image input. For this report both of these were used. Images in these intermediary programs can be scaled to the HyperCard frame. Illustrations and photographs for example, can be "cleaned-up", while contrast and brightness can be adjusted preferentially. When a satisfactory image is completed the image is stored in either a RIFF, or PICT 2 format and transferred to Hypercard. Other storage formats are possible, but unfortunately they tend to store more information than HyperCard can handle or use a language that is unreadable by HyperCard, such as PostScript.

With the image placed onto the Hypercard "card", buttons on the image itself or on the right hand column of buttons can be programmed to move the reader through the building room by room. (Fig.4)

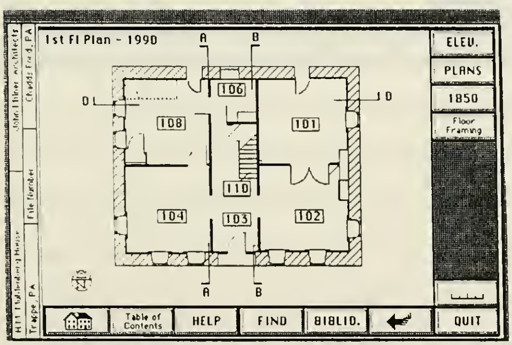


Figure 4. This sample plan indicates the many options available to the program user.

In addition, properly designed buttons can draw the reader back to a portion of the text relevant to the screen image. (Fig. 5 - 6) (next page)

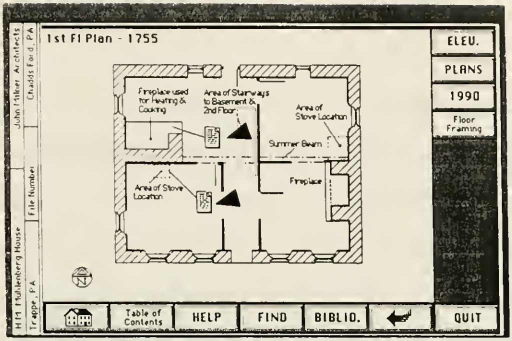


Figure 5. Arrows indicate "buttons" that when clicked, can refer the user to relevant text information.

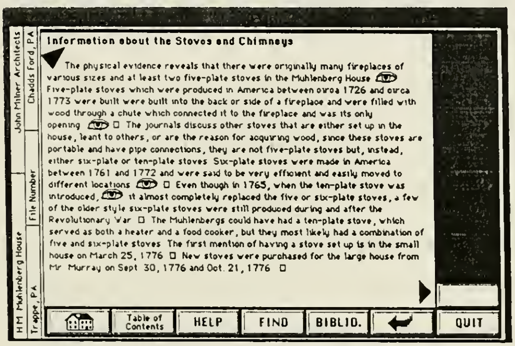


Figure 6. This is a sample of the text that is accessed when "buttons" such as those above are activated.

In this project, all of these connections through buttons have been demonstrated. Once again the card file library drawer serves as a good analogy. If each card on which we have entered some type of information is assigned its own particular number we can then begin to cross reference cards by including the numbers of related cards on each. If, for example, I pick out the card detailing the use of five- six- and ten plate stoves in the house, it might be marked with the numbers of other cards which I can refer to for Photographic or illustrative information. These cards in turn might refer me to the numbers of cards that show room plans and conjectural locations of the stoves. These can refer to other cards and so on and so on. This is exactly the basic programming system in use in the project demonstrated.

CONCLUSIONS

Information flow, and the quest to control it, has been desire of civilizations spanning many thousands of years. The computer, devised during the past half century has brought this desire closer to reality than ever before. This is especially true in the research disciplines that require the painstaking gathering of minutiae whose assembly into a coherent whole depends upon the successful assimilation and relation of vast quantities of data.

Like many sciences, research and data gathering are the essential tools of the historic preservation practices. During the research process, many bits of information are collected. Information may be an undated photo, a diary entry, a chip of molded wood or an entire dressed sandstone facade. Each preservationist/researcher is driven to take these informational bits and present them as related threads in the historical fabric of a structure. The presentation to date, however sophisticated, has always relied upon the conventional book format. Relationships between material presented on page 18 and material presented on page 318 or 418 were not always clear. The book format dictated the arrangement of material cross-referencing, while possible, would have added considerable time to the effort and bulk to the text.

Aside from being able to calculate extraordinarily well, desktop (and even laptop) computers excel at searching for and finding a single piece of stored data among literally millions of other pieces. It is this characteristic of their design that we rely upon to search out a book in a collection of millions. It is also this ability that we rely upon to verify our medical insurance number during a hospital admission. It is also this ability that we can use to

harness the data that are collected during the research portion of the Historic Structure Report.

When an ordinary desktop computer is operating with an organizational program such as HyperCard, an entirely new dimension of data storage, retrieval and presentation is open to us. All of the information gathered can be stored, and relationships can be developed among the elements of the database with a flexibility never available in the past. It is the development of these relationships that fosters a greater understanding of the gathered information.

THE FUTURE

The demonstration presented here is only a glimpse of the technology that is currently available. HyperCard, even as it is currently designed, can perform two very important tasks that were not taken advantage of for this project.

Even though HyperCard presents a clean, but low resolution image on the screen (memory conservation), it has the ability to search out a large format, high resolution images created in a number of other programs. Computer aided design (CAD) images can be located and printed through HyperCard commands. If an architect wishes to save and reproduce full scale drawing in conjunction with an HSR, he is not limited to the screen image.

In addition to this, HyperCard is designed to control compact and full size video discs. A full size (12") disc will hold approximately 55,000 images. Consequently one disc might contain photographs of a number of projects. When the right "button" is activated, A window (scalable) will open on the

main screen and the desired photograph may be shown. In addition, videodiscs (and even some magnetic disks) can store short movies that might include a video tour through the property or the chief steps in a conservation process. These movies may appear in an on-screen window as easily as a single photograph.

Finally, computer management of HSR's lays the groundwork for a large scale national (international?) database of reports. If they are broken down into organizational units as I have proposed in this study, data access and comparison among hundreds of other reports would be almost instantaneous. Obscure wallpapers or renders, for example, could be searched out and compared to similar materials in a current project. Data gathering could build upon the work of many, and hence, redundancy of effort could be reduced. The advantages of a computer managed database are unquestioned. The preservation/conservation practices would do well to begin to plan with this goal in mind. Much is being done already, but like so many weavers encircling a carpet, an early plan and communication will go a long way toward determining the final product.

BIBLIOGRAPHY

Apple Computer. *HyperCard Stack Design Guidelines.* New York: Addison-Wesley, 1989.

Burke, James A. *The Day the Universe Changed.* Boston: Little Brown and Co.,1985

Claris Corporation. *HyperCard Script Language Guide.* Apple Computer Corporation, 1989 - 1990.

Goodman, Danny. *The Complete HyperCard Handbook.* 2nd ed. Toronto:Bantam Books, 1988

_____________. *HyperCard Developer's Guide.* Toronto: Bantam Books, 1988.

Fraase, Michael. *Macintosh Hypermedia.* Vol.1 *Reference Guide.* Glenview, IL: Scott, Foresman and Co., 1989.

Heid, Jim and Peter Norton. *Inside the Apple Macintosh.* New York:Brady Publishing, 1989.

Kapsch, Robert. "HABS/HAER: A User's Guide." *APT Bulletin* 22, no. 1&2 (1990): 22-34.

Landow, G. "The Rhetoric of Hypermedia: Some Rules for Authors."*Journal of Computing in Higher Education* 1 no. 1 (1989): 39-64.

Lynch, Patrick. "Stacking Books: Adapting Print Projects to Multimedia." *Syllabus* 11 (May/June 1990): 20-21

Nelson, Ted. *Computer Lib/Dream Machines.* Redmond,WA: Microsoft Press, 1987.

Pile, James E. and Joseph R. Pierce. "Modifying HyperCard's inheritance path." *Inside HyperCard* 1, no. 3 (May 1991): 1,14-16

Salkind, Neil J. *The Big Mac Book.* 2nd ed. Carmel: QUE Corporation, 1991

Winkler, Dan and Scott Knaster. *Cooking with Hypertalk 2.0.* New York: Bantam Computer Books, 1990.